PATH PLANNING OF MOBILE ROBOT WITH GROWING NEURAL GAS AND ANT COLONY OPTIMIZATION

Thiago Azevedo, Claudio César, Allan C. Gomes, Marcus Davi, Thiago A. Lima, Vinícius R. A. Ferreira, Wilkley B. Correia, Arthur P. S. Braga *

** Universidade Federal do Ceará*
Departamento de Engenharia Elétrica
Grupo de Pesquisa em Aumotação, Controle e Robótica

Email: `thiagoaccosta@hotmail.com`, `claudiofilho22@gmail.com`, `acg@alu.ufc.com`, `davi2812@dee.ufc.br`, `thiago.lima@alu.ufc.br`, `viniciusdosreis@hotmail.com`, `wilkley@dee.ufc.br`, `arthurp@dee.ufc.br`

Abstract— This paper presents a new path planning strategy obtained from the combination between Growing Neural Gas (GNG) and Ant Colony Optimization (ACO) algorithms. The proposed strategy was tested in a real mobile robot for two different scenarios and compared with the APF approach. Positive and negative points of the proposed strategy are highlighted throughout the text. As one positive point, the proposed path planning strategy presented a better end positioning of the robot in comparison with APF in the performed tests - an aspect of great interest for practical applications in industry and medical area.

Keywords— Self-Organizing Maps, Metaheuristics, Artificial Intelligence, Path Planning, Mobile Robot

1 Introduction

The field of robotics has grown in recent years, as it became crucial to a wide diversity of applications. To cite a few, mobile robots are used in the manufacturing industry (where robots transport loads and assemble parts) and in the medical area, with robots acting in medicine manipulation at pharmaceutical labs and even performing supervised actions in surgery rooms (Dogangil, 2010).

Motivated by such needs, autonomous robots are crucial to execute tasks where machines must be able to read data from an environment and produce adequate navigation routes. However, in order to achieve this goal, it is mandatory to use techniques that are able to classify areas between occupied and available free spaces, so that the robot can autonomously avoid collisions. Thus, getting a map that represents the environment workspace is the very first needed in order to implement robot navigation algorithms. In this respect, different types of sensors can be adopted for 2D and 3D mapping (Correa, 2013).

Several path planning strategies have been employed in the field of mobile robotics as surveyed by (Ottoni, 2000). To cite a few, both (Yao et al., 2010) and in (Wang et al., 2011) apply techniques based on A* Algorithm, with the later specifically using the Dijkstra Algorithm. There is a group of techniques that base themselves with Metaheuristic, such as the strategy proposed in this paper. On what concerns Methaeuristic strategies, some variations have been presented by using a merge between two techniques like the joint between Ant Colony Optimization (ACO) and the Artificial Potential Fields Approach (APF) in (Mei et al., 2006). Combination between ACO and Genetic Algorithms has also been studied by (Châari et al., 2012).

That way, this paper objective is to expose and test a strategy for path planning in real time environments based on two techniques: the Growing Neural Gas (Fritzke, 1995) and Ant Colony Optimization (Stützle and Hoos, 2000) algorithms. The implemented strategy is compared with an already established strategy, the Artificial Potential Fields Approach (Goodrich, 2002). Experimental results for both strategies are presented.

The paper content is split in six sections. After the introduction, Section 2 describes the structure of the mobile robot in terms of both software and hardware. In Section 3 the Growing Neural Gas (GNG) is explained while details on its implementation are presented. In Section 4, the Ant Colony Optimization (ACO) is presented. In Section 5, the results are presented and comments about the experiments are given to establish a comparison between Artificial Potential Field (APF) and the proposed fusion of GNG with ACO. Conclusion remarks are finally brought in the last section of this paper.

2 Mobile Robot Description

A real mobile robot was used to evaluate the path planning and reference tracking of the proposed strategy. The mobile robot possesses geared DC motors that drive two differential wheels, while other two smaller passive wheels are responsible for the mobile robot equilibrium.

The planning algorithm is implemented in an embedded Linux Raspberry Pi 3 system running the ROS (Robotic Operating System) environment (Quigley et al., 2009). The ROS was used to handle messages between the Raspberry Pi and

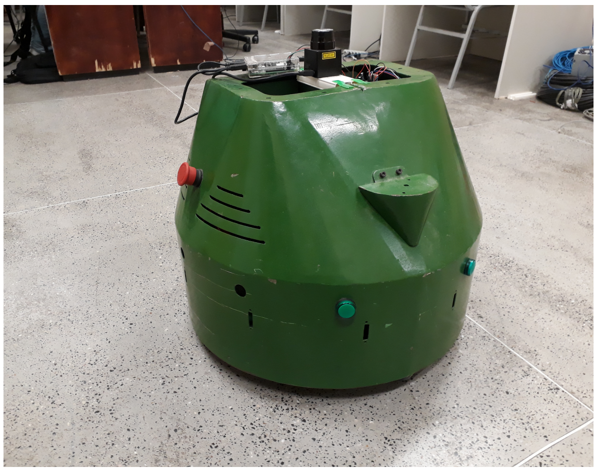

Figure 1: Mobile Robot

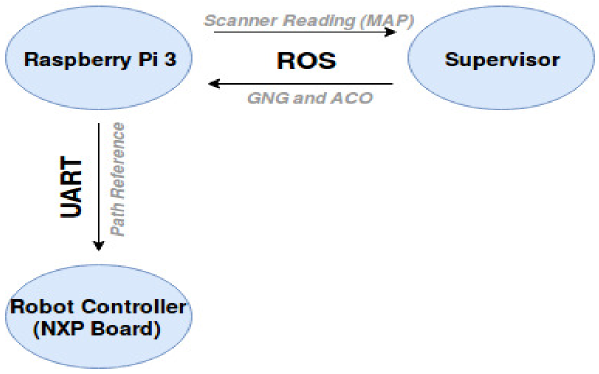

Figure 2: Hardware Diagram

a supervisory computer on which both GNG and ACO algorithms were taking place. The Raspberry Pi was responsible for communicating with a Hokuyo Laser Scanner and constructing a topological map of the experimental domain. Different settings of obstacles were tested for evaluating the algorithms performance.

The motor control and path tracking was implemented in the NXP FRDM-K64F development board, which received the reference path at each time from the Raspberry Pi. Non-Linear control strategies were used (Klancar et al., 2005) to correct the mobile robot path along the planned reference. The motors were controlled using digital PI controllers.

The ROS environment (Robot Operating System) is a framework system that support the development of robotics applications, and a collaborative community (Quigley et al., 2009). In the platform, ROS was used as a way of reading the Laser Scanner data with functions of the package (Kohlbrecher et al., 2011) that handles the scanner signal to create a map of the room around the robot, and that feeds the algorithms implemented by the authors. In general lines, ROS was the bridge to communicate the scanner connected on a Raspberry PI 3 and a supervisor computer, that was responsible to execute the algorithms. Once executed, the planned path is sent back to the Raspberry Pi, which in turn transmits the information to the NXP FRDM-K64F microcontroller by UART communication. Figure 2 represents the complete hardware system and how they communicate to each other.

3 Growing Neural Gas (GNG)

Once the Hokuyo Laser Scanner reads the environment around the robot, the result is a matrix of points containing information about the occupancy of the spots read by the scanner, including free spaces, occupied spaces and uncertain spaces. Depending on the resolution of the reading, it could be thousands of points for the robot to process.

In order to simplify the Path Planning goal in computational aspects, the Growing Neural Gas (Fritzke, 1995) application is a way of reducing the complexity of the reading without losing its primary properties, like dimensions and free spaces where the robot can navigate, generating a topological map.

The main goal of using GNG is to reduce drastically the number of points of interest, once it will only focus on representing free spots, avoiding occupied spaces and preserving the original shape of the map. The number of points that has to be analyzed, that once was around the thousands of points, is now around 100 points, depending on the parameters used in the algorithm.

The algorithm works receiving a probability density function $P(\xi)$, based on the points of the original read. It starts by randomly inserting two nodes in the current created space, and continues by moving nodes, creating or deleting edges between nodes according to $P(\xi)$, which evolves based on previous determined parameters exposed as follow.

The main idea of the algorithm is to update the nodes position under influence of the real reading, our probability density $P(\xi)$. A very important parameter is λ, because it controls the period of iterations when nodes are created. In every $k\lambda$ iteration (with k = 1, 2, 3, ..., n) a new node is created between the node with the biggest accumulated error and its neighbor with biggest accumulated error. Hence, the greater is λ, the greater is the time that the algorithm reallocates its network before adding new units. There is a list containing the parameters.

- λ = controls node creation;
- sizemax = controls the maximum number of nodes;
- agemax = controls the maximum age of edges;

- α = factor that changes error of nodes in kλ iterations;
- e_b, e_n = moves nodes through iterations;
- d = factor that decreases error of all nodes.

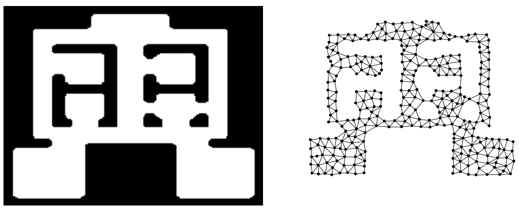

Figure 3: GNG Graph

Figure 3 shows the result of applying GNG in a theoretical space (Braga, 2004), which is a network graph containing several nodes, less than in the original read, with edges connecting close nodes and a great similarity with the space itself, in a very simplified way. It is now given several paths connecting every pair of nodes in the graph. It is the Ant Colony Optimization (ACO) algorithm duty to recognize the path between two nodes in the graph that would take less time for the robot to perform, hence, the better path.

The space that will base the GNG algorithm and that the robot will navigate is not actually theoretical, is a real time reading from the Laser Scanner of the space where the robot is. Figure 4 is the result of GNG applied on a real laser reading of a test room. The obstacles were augmented by $0.3m$ on software to feed GNG, in order to cover the robot's radius and guarantee that it can navigate safely (in Figure 4 GNG is overlaid in the scanner reading without augmentation, in real size).

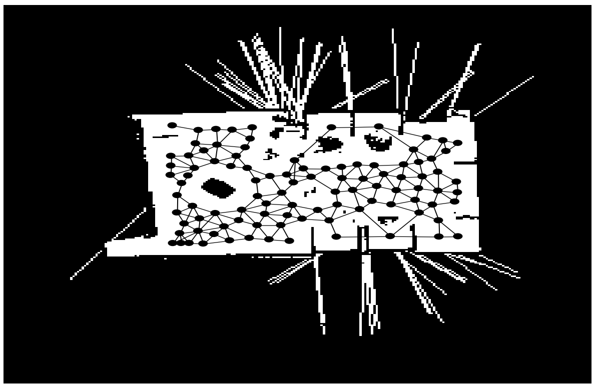

Figure 4: GNG Graph based on LIDAR reading

Notice that GNG network does a great representation of the room, displaying available spots where the robot can navigate and connecting it so it can give multiple paths from some point to another. The test room was a complex space, with different kinds of obstacles in several sizes, including some very thin ones. That is the advantage of using a laser scanner allied with GNG. Even in complex scenarios with a high difficult level the robot can safely navigate.

4 Path Planning Strategy with ACO

Following the generation of the the topological map by the GNG, a graph $\boldsymbol{G(N,E)}$ that represents the environment in which the robot is inserted is obtained. Disregarding the kinematics and dynamics constraints and since the GNG only uses the free space data to construct the topological map, which guarantees the absence of obstacles in the generated graph, the path planning problem comes down to finding an optimal or sub-optimal path between two points in a graph.

For this task it was used the meta-heuristic ACO (Dorigo and Stützle, 2004), which provides a framework to solve discrete optimization problems through the use of artificial ants. Those ants move randomly between the nodes and across the edges of a graph. They use the data of the amount of pheromone deposited by the ants themselves in the edges of the graph as basis for the decision making process of the movement.

The ant algorithm used was the MAX-MIN Ant System (MMAS) (Stützle and Hoos, 2000) with adaptations to improve its performance regarding the task of finding short paths in a graph. MMAS algorithms insert maximum τ_{max} and minimum τ_{min} limits for pheromone values, thus avoiding a quick convergence to a sub-optimal path and encouraging greater exploitation by the ants.

4.1 Algorithm Description

At the beginning of the algorithm a total number of M ants are created having as the initial position s, which is the location where the robot is. In addition, the same pheromone value τ_0 is given to all the edges in the graph $\boldsymbol{G(N,E)}$. From this, each ant start the construction of the solution to the destination node t randomly. The probability P_{ij}^m of an ant m that is on the node i moves to the neighbor node j, which belong to the neighbor node group of i, called J, through the edge that connects i to j, is given by Equation 1:

$$P_{ij}^m = \frac{[\tau_{ij}]^\alpha[\eta_{jt}]^\beta}{\sum_{j\in J}([\tau_{ij}]^\alpha[\eta_{jt}]^\beta)} \qquad (1)$$

Where τ_{ij} is the amount of pheromone deposited on the edge that connects i to j, η_{jt} is a parameter based on the available heuristic information, which in this case is the inverse of distance between node j and the destination node

t, measured by the Manhattan metric, given by Equation 2, where x and y are the coordinates of i and j in each axis:

$$H_{jt} = \frac{1}{|x_j - x_t| + |y_j - y_t|} \tag{2}$$

The parameters α and β are used to determine, respectively, the level of contribution the pheromone trail and the heuristic information has on the decision making process of the ant.

During the construction of the solution the ants store the path traversed, eliminating the loops that can occur during this process. After all the ants reach the destination, the pheromones are updated according to the best path obtained by ant f and the evaporation rate ρ according to the Equation 3:

$$\tau_{ab}^{f} = \rho\tau_{ab}^{f} + \Delta\tau \tag{3}$$

Where τ_{ab}^{f} represents all the arcs present in the path traversed by ant f and $\Delta\tau$ represents the amount of pheromone to be deposited, calculated by the inverse of the euclidean distance $dist_{st}^{f}$ traveled by the ant f between the starting point s and the destination point t (Equation 4):

$$\Delta\tau = \frac{1}{dist_{st}^{f}} \tag{4}$$

After the update of the pheromones, it is verified if they are within the limits imposed by the maximum and minimum parameters, and they are updated according to Equation 5. The parameters τ_{max} e τ_{min} are obtained by Equations 6 e 7 (Pellegrini et al., 2006). Where C_{best} is the euclidean distance of the best path found so far, and n is the number of nodes from the graph $\boldsymbol{G(N,E)}$:

$$\tau = \begin{pmatrix} \tau_{min} & if\ \tau \leq \tau_{min} \\ \tau & if\ \tau_{min} \leq \tau \leq \tau_{max} \\ \tau_{max} & if\ \tau \geq \tau_{max} \end{pmatrix} \tag{5}$$

$$\tau_{max} = \frac{1}{\rho C_b est} \tag{6}$$

$$\tau_{min} = \frac{\tau_{max}(1 - \sqrt[n]{0.05})}{(\frac{n}{2} - 1)\sqrt[n]{0.05}} \tag{7}$$

The last part of the algorithm is the convergence test. For MMAS algorithm the notion of convergence is defined in (Stützle and Hoos, 2000) as follow: "The MMAS has converged if for each choice point, one of the solution components has τ_{max} as associated pheromone trail, while all alternative solution components have a pheromone trail τ_{min}" (p. 899). In case the algorithm takes too long to converge, a parameter number of iterations I has been inserted, so that the algorithm ends if it reaches this limit.

4.2 Aspects of the Practical Implementation

To avoid problem of numerical order, such as negative numbers and magnitude discrepancies, at the beginning of the algorithm, the coordinates of the map generated are normalized between 0 and 1 according to the dimension of the map. Before the end of the algorithm the final path is re-scaled to the original scale of the map.

The algorithm requires the following parameters:

- The Graph $\boldsymbol{G(N,E)}$ given by the GNG algorithm;
- The current and the target position (s and t) of the robot;
- Data of resolution and dimension from the map, given by the ROS package hector-mapping.

By the start of the algorithm the following parameters must be initiated:

- Number of the ants M;
- Number of iterations;
- Evaporation rate ρ;
- α and β parameters.

The pheromones related parameters τ_{max}, τ_{min} and τ_0 are started with the value of 1.

5 Experimental Results

In order to experiment the proposed path planning strategy and for comparative purposes, an already established strategy was implemented, the Artificial Potential Field Approach, which is a simple and very used path planning approach that presents great results without extreme computational processing (Goodrich, 2002) and (Qureshi and Ayaz, 2016). This technique is based on the behavior of a charged particle in a magnetic field, and its actions on the environment depends on the combination of the attractive fields (influenced by the destination points) and the repulsive fields (represented by the obstacles on the map). The Artificial Potential Fields algorithm implemented used these parameters:

- r = target radius;
- a = attractive field scalar factor;
- robs = obstacle radius;
- b = scalar factor of the repulsive field;
- s = field propagation factor;
- λ = less distance increment.

The tests were made in two different scenarios, both in a closed room with obstacles differently placed in space and fixed starting and finishing positions (Figures 5 and 7). The obstacles are the walls and the objects deliberately placed to compose the scenario. With the purpose of not hitting the obstacles properly, all of them were augmented on software by $0.3m$, which is the radius of the robot, so the map can feed both strategies safely. The robot navigates with constant speed of 0.1 meters per second for both algorithms. The first scenario is composed with only two objects to avoid, with the intuition to stimulate well curved paths. The second scenario is composed with three objects making a bifurcated way through to the finishing position. The robot traveled (manually) in the room to read the scenario around it with the LIDAR sensor and, after that, it came back to the origin point, compiled all the informations and executed its path to the finishing point sufficiently close to the exit door, selected by an operating person in real time. The real path was measured with the robot odometry.

As values of comparison between the two strategies in both scenarios, it was measured computational time (t), total distance traveled (D), the difference between the real ending position and finishing point (d_f) and an MSE (Mean Squared Error) of θ, the orientation of the robot, in order to analyze the smoothest path, which is the path with less mean variations in θ. The $MSE(\theta)$ measure is taken as follow, with N being the number of θs:

$$MSE(\theta) = \frac{1}{N-1}\sum_{i=1}^{N-1}(\theta(i+1)-\theta(i))^2 \qquad (8)$$

The path supplied by the path planning algorithm was controlled using a non-linear approach (Klancar et al., 2005). The controller is tuned by linearizing the mobile robot model and allocating the system poles so that it remains stable while also specifying performance parameters. The motor speed is controlled using a PI controller whose gains were tuned to allow fast response and eliminate steady-state error for each motor velocity references. Since the reference tracking is computed in a digital micro-controller, it is required to specify a reasonable sampling time for the closed-loop performance. A sampling time of 10 Hz for reference tracking closed-loop exhibited good performance considering the mobile robot dynamics.

These are the parameters for all algorithms used:

$GNG:$	$ACO:$	$APF:$
$sizemax = 60$	$epoch_{max} = 50$	$r = 0.03$
$agemax = 100$	$n_{ants} = 50$	$robs = 4$
$\lambda = 300$	$evapfactor = 0.98$	$a = 5$
		$b = 5$
		$s = 18$
		$\lambda = 0.03$

In order to make explanation easier to understand and more organized, GNG and ACO will be called Strategy A and the Artificial Potential Fields approach will be called Strategy B. It is important to affirm that the ending and starting points for both algorithms are the same for Scenario A and Scenario B, in order to have a fair comparison between the two techniques.

5.1 1st Scenario

The first scenario was composed by two deliberated obstacles, one close to the robot's starting position and the other close to the finishing position. Intuitively, the greatest path is not difficult to achieve, but will result in some curved way while exploring the room and that could represent greater variations on θ. Figure 5 represents this scenario.

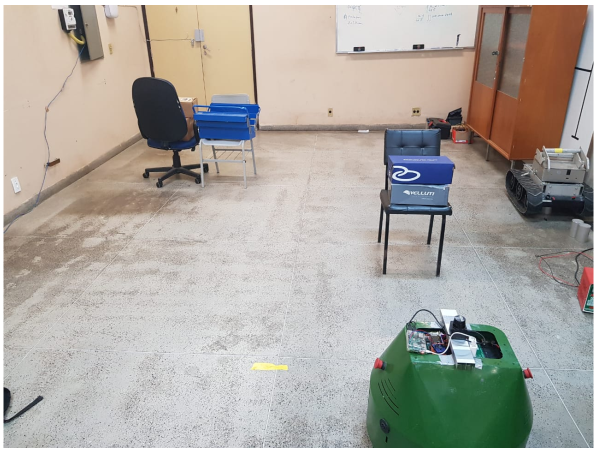
Figure 5: 1st Scenario

In this scenario it was possible to notice that both algorithms were able to execute very well the path proposed. In the collected results exposed in Table 1 the conclusions were very tight, with the Strategy A presenting a longer processing time and a longer path, with a difference of 0.40m. Also, Strategy A had a bigger $MSE(\theta)$, representing a path with more orientation changes. In the precision parameter $d_f(m)$ the difference was also very slight, with a really smooth advantage for Strategy A.

Table 1: 1st Scenario's Results

	GNG and ACO	APF
$t(s)$	0.9643	0.0621
$D(m)$	6.091	5.686
$d_f(m)$	0.071	0.123
$MSE(\theta)$	0.0062	0.0007

The reading of the Laser Scanner in the actual test such as the curves of reference path and real executed path are available to see in Figure 6.

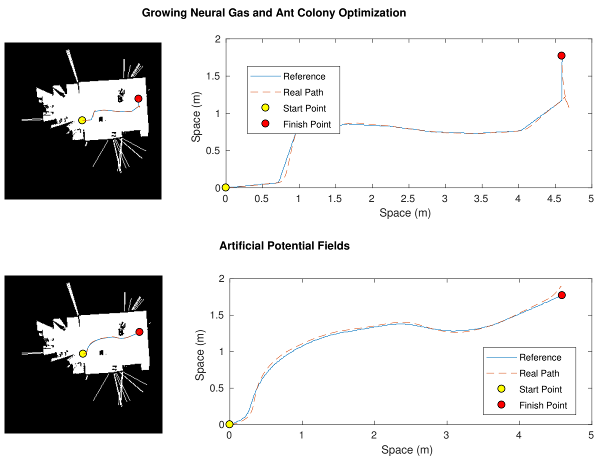

Figure 6: 1st Scenario's Paths

These curves are important to see how the controller behaved in the proposed paths. Strategy A have some points with high θ variation, resulting in a bad behavior of the controller, that has a little delay to go back to the right path.

5.2 2nd Scenario

The second scenario was composed by three obstacles, configuring together a bifurcated path. Now the complexity of the problem had increased, since now there is more than one possible solution for the problem, with a higher obstacle density and narrow paths, a challenge for the robot to go through without hitting the anything. Figure 7 represents this scenario.

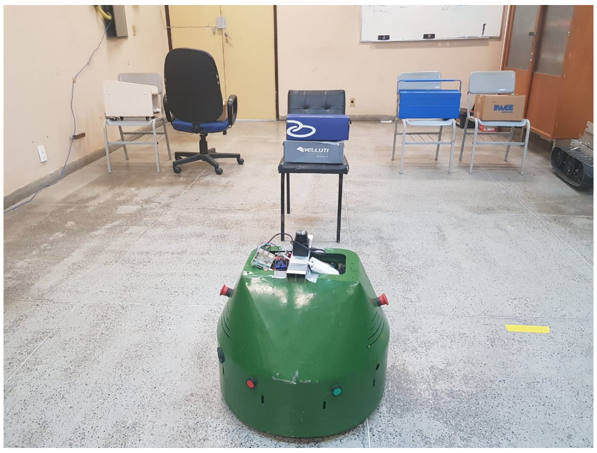

Figure 7: 2nd Scenario

In this scenario it was possible to observe a typical APF problem, the local minimum problem. Usually it occurs due to a great repulsion that the obstacles cause denying the robot to achieve its objective, because before it reach the goal position, attractive and repulsive fields become with extreme close module values, causing a resulting field equal to zero.

While Strategy B was impaired by this issue, Strategy A behaved very well, achieving the goal position and avoiding all obstacles, although the narrow spaces. Observing Table 2, Strategy A again presented a larger computational time, a higher $MSE(\theta)$ and a really higher precision measured by $d_f(m)$, due to that Strategy B limitation, which do not effect at all Strategy A, since it has no local minimum problems.

Table 2: 2nd Scenario's Results

	GNG and ACO	APF
$t(s)$	0.9936	0.0935
$D(m)$	5.980	3.003
$d_f(m)$	0.014	2.675
$MSE(\theta)$	0.0079	0.0016

Finally, the reading of the laser scanner and the curves of reference path and real path are available in Figure 8 to see how well the controller behaved.

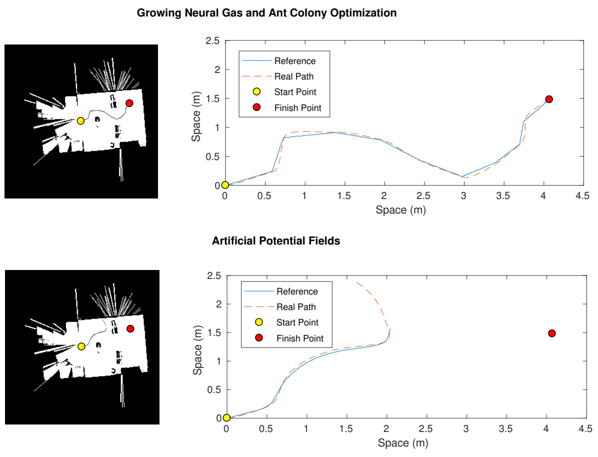

Figure 8: 2nd Scenario's Paths

Again the Strategy A path has suffered with some punctual high variations in θ that made the controller have some difficult while executing it.

6 Conclusions

Results obtained from experiments in both scenarios lead to the conclusion that the proposed combination of Growing Neural Gas allied with Ant Algorithm Optimization is a viable strategy for path planning with a better end positioning of the robot in comparison with APF in the performed tests (Tables 1 and 2). While the APF strategy had less time of processing and a very well performance in the first scenario, the strategy proposed had satisfactory performance in both scenarios and delivered the robot to its location safely, while avoiding obstacle collision even when navigating in narrower paths and when APF could not. Such characteristics make adaptation for real spaces and different kinds of obstacles prospective and indicate potential for use in industrial applications.

In addition, further exploration of the proposal is expected in order to improve crucial points in the path planning algorithm. As an example, application of a smoothing technique could make sure that extreme variations on the robot orientation angle θ are avoided.

A deeper attention on dynamic environments is an action moving forward with the studies, once the strategy presented isn't capable of dealing with dynamic environments. Therefore, that approach is a good advance to be treated in future works.

References

Braga, A. P. S. (2004). Agente topológico de aprendizagem por reforço, *Tese de Doutorado. Engenharia Elétrica*, Universidade de São Paulo.

Châari, I., Koubâa, A., Bennaceur, H., Trigui, S. and Al-Shalfan, K. (2012). smartpath: A hybrid aco-ga algorithm for robot path planning, in: In 2012 IEEE Congress on evolutionary Computation (CEC), Brisbane, Australia, pp. 1–8.

Correa, D. S. O. (2013). *Navegação autônoma de robôs móveis e detecção de intrusos em ambientes internos utilizando sensores 2D e 3D, Instituto de Ciências Matemáticas e de Computação*, PhD thesis, Universidade de São Paulo, USP.

Dogangil, B. L. Davies, R. y. B. F. (2010). A review of medical robotics for minimally invasive soft tissue surgery, *Proc. Inst. Mech. Eng. Part H: J. Eng. Med., vol. 224, no. 5, pp. 653–679.*

Dorigo, M. and Stützle, T. (2004). *Ant Colony Optimization*, Bradford Company, Scituate, MA, USA.

Fritzke, B. (1995). A growing neural gas network learns topologies, *Advances in neural information processing systems*, pp. 625–632.

Goodrich, M. A. (2002). Potential fields tutorial.

Klancar, G., Matko, D. and Blazic, S. (2005). Mobile robot control on a reference path, *Proceedings of the 2005 IEEE International Symposium on, Mediterrean Conference on Control and Automation Intelligent Control*, pp. 1343–1348.

Kohlbrecher, S., Meyer, J., von Stryk, O. and Klingauf, U. (2011). A Flexible and Scalable SLAM System with Full 3D Motion Estimation, in IEEE International Symposium on Safety, Security, and Rescue Robotics, Kyoto, Japan.

Mei, H., Tian, Y. and Zu, L. (2006). A hybrid ant colony optimization algorithm for path planning of robot in dynamic environment, Int. Journal of Information Technology. 12(3): pp. 78-88.

Ottoni, G. (2000). Planejamento de trajetórias para robôs móveis, Projeto de Graduação apresentado ao Curso de Engenharia de Computação, Fundação Universidade Federal do Rio Grande, Rio Grande, RS.

Pellegrini, P., Favaretto, D. and Moretti, E. (2006). On max-min ant system's parameters, *Ant Colony Optimization and Swarm Intelligence*, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 203–214.

Quigley, M., Gerkey, B., Conley†, K., Faust, J., Foote†, T., Leibs, J., Berger, E., Wheeler, R. and Ng, A. (2009). Ros: an open-source robot operating system, in Open-source software workshop of the Int. Conf. on Robotics and Automation, Kobe, Japan.

Qureshi, A. H. and Ayaz, Y. (2016). Potential functions based sampling heuristic for optimal path planning, Auton Robot 2016; 40: 1079–1093.

Stützle, T. and Hoos, H. H. (2000). Max-min ant system, Vol. 16, Elsevier Science Publishers B. V., Amsterdam, Netherlands, pp. 889–914.

Wang, H., Yu, Y. and Yuan, Q. (2011). Application of dijkstra algorithm in robot path-planning, *2011 Second International Conference on Mechanic Automation and Control Engineering*, pp. 1067–1069.

Yao, J., Lin, C., Xie, X., Wang, A. J. and Hung, C. C. (2010). Path planning for virtual human motion using improved a* star algorithm, *2010 Seventh International Conference on Information Technology: New Generations*, pp. 1154–1158.